



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|-----------------|-------------|----------------------|---------------------|------------------|
| 09/449,782      | 11/26/1999  | JAMES MCKEETH        | MICE-0089           | 6698             |

7590

08/07/2003

COE F MILES  
TROP PRUNER HU & MILES P C  
8554 KATY FREEWAY  
SUITE 100  
HOUSTON, TX 77024

EXAMINER

STEELMAN, MARY J

ART UNIT

PAPER NUMBER

2122

DATE MAILED: 08/07/2003

11

Please find below and/or attached an Office communication concerning this application or proceeding.

3



UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents  
United States Patent and Trademark Office  
P.O. Box 1450  
Alexandria, VA 22313-1450  
[www.uspto.gov](http://www.uspto.gov)

**BEFORE THE BOARD OF PATENT APPEALS  
AND INTERFERENCES**

Paper No. 11

Application Number: 09/449,782

Filing Date: November 26, 1999

Appellant(s): MCKEETH, JAMES

Indranil Chowdhury, Reg. No. 47,490  
For Appellant

**MAILED**  
AUG 07 2003  
Technology Center 2100

**EXAMINER'S ANSWER**

This is in response to the appeal brief filed 06/27/2003.

**(1) *Real Party in Interest***

A statement identifying the real party in interest is contained in the brief.

Art Unit: 2122

**(2) *Related Appeals and Interferences***

A statement identifying the related appeals and interferences which will directly affect or be directly affected by or have a bearing on the decision in the pending appeal is contained in the brief.

**(3) *Status of Claims***

The statement of the status of the claims contained in the brief is correct.

**(4) *Status of Amendments After Final***

No amendment after final has been filed.

**(5) *Summary of Invention***

The summary of invention contained in the brief is correct.

**(6) *Issues***

The appellant's statement of the issues in the brief is correct.

**(7) *Grouping of Claims***

Appellant's brief includes a statement that claims 1-22 do not stand or fall together and provides reasons as set forth in 37 CFR 1.192(c)(7) and (c)(8).

**(8) *Claims Appealed***

The copy of the appealed claims contained in the Appendix to the brief is correct.

**(9) *Prior Art of Record***

6182279      Buxton      01-2001

**(10) *Grounds of Rejection***

The following ground(s) of rejection are applicable to the appealed claims:

Art Unit: 2122

Claims 1-22 are rejected under 35 U.S.C.102(b). This rejection is set forth in prior Office Action, Paper No. 5.

**(11) Response to Argument**

Appellant's arguments regarding claims 1-22 are as follows:

A. Claims 1, 15 and 21 recite "receiving an identifier, receiving output from a command line utility, and storing the command line utility output in system storage at a location identified by the identifier." Buxton does not disclose receiving output from a command line utility and storing the command line utility output in a system storage at a location identified by a received identifier. (See page 8, last paragraph.) Buxton does not disclose receiving output from a command line utility, but rather teaches that the user, through a GUI and editor, inputs customizations to the base component. (See page 10, lines 10-12.) Storage of a template in a template storage file is not storage of command line utility output in system storage. (See page 10, lines 15-16.)

B. Claim 2 depends from claim 1 and adds the act of "receiving an identifier comprises receiving an identifier that identifies one or more entries in a system registry database." Buxton does not disclose receiving an identifier that identifies one or more entries in a system registry database but rather creation of registry keys for spoofing of templated components. (See page 11, second and third paragraphs.)

C. Claim 12 depends from claim 1 and adds the "act of storing comprises associating each line of command line utility output with a line identifier in the system storage." Buxton does not disclose that the act of storing comprises associating each line of command line utility output with a line identifier in the system storage. (See page 12, second paragraph.)

Examiner's Response to the Appellant's arguments is as follows:

A. Buxton (col. 8., lines 51-53) provides command line processes as an alternative to a user interface. At col. 12, lines 2-3, Buxton recites, "User may enter a descriptive name of the template..." Thus, name becomes a property (an identifier) of the template component. At col. 12, lines 4-7, Buxton recites, "Template (component) Properties menu option generates a listing of the template name, the date of creation...the date of most recent modification...and a descriptive name of the template (identifier)." Col. 13, lines 8-10 further recite, "Template builder 400 creates templates 420...with the assistance of the template storage DLL 205." Also see col. 12, lines 43-49, which state, "Editor module 404 of template builder 400 enables users to selectively modify the functionalities contained within a base component such as component 300 or FIG 3A...an original component is opened, changes are made using editor 404, and the modified component is saved as a template. That is to say, the modified component, storing the command line utility output or output from the command line utility (the modified component) is saved and/or stored. Col. 13, lines 10-14, further recite, "Template storage DLL (system storage) performs a number of formatting and storage/retrieval methods which manage the storage and registration of templated components..." See Figure 2, block 212 and associated text at col. 7, lines 48-67 and col. 11, lines 17-18, further recite, "...templates may be stored to facilitate efficient distribution..."

Thus, Examiner does show that Buxton discloses receiving the modified component output from a command line utility and storing the command line utility output (the modified component) in system storage at a location identified by a received identifier. Furthermore,

Art Unit: 2122

storage of a modified template component in a template storage file is “storing command line utility output in a system storage”.

B. Buxton discloses receiving an identifier that identifies one or more entries in a system registry database. Col. 9, lines 5-7, recite “Components register themselves in registry 250 with WIN32 APIs 240 so that the OLE libraries 230 are aware of the component’s existence.” Col. 13, lines 14-19 further recite, “Template storage DLL 205 ensures all additional registry keys and library types are created for template 420...” The paragraph explains that information is used (identifies one or more entries in a system registry database) by the OLE libraries and the base controller container. How the identifier registry database is used (in this case to spoof a template component to appear to be a base control of the original component) is not a claim limitation. Also see col. 14, lines 1-4, which recite, “Referring to FIG. 4B, each template 420 comprises initialization data 425, user instructions 445, and one or more registry keys or subkeys 450.” And col. 14, lines 9-11, which recite, “Additional data to identify and register the component is also saved in various ISTREAMS and ISTORES in the templates ISTORE” Additionally, col. 14, lines 20-59 provides more information on the identifiers. Specifically, lines 43-48, which state, “Key 450 G contains information indicating the CLSID of the object upon which the subject template is based. Key 450 H contains information indicating the name of the storage object in template storage file 440...” Also see figure 2, block 250. The component loader uses the identifier. The component loader uses saved information “i.e. the initialization information 425, user instruction 445 and key 450” of the component to wrap / spoof the base component and create a composite object. (col. 13, line 16 and col. 21, lines 58-67). Thus, Buxton does show

Art Unit: 2122

that the system uses an “identifier that identifies one or more entries in a system registry database.”

C. Buxton discloses (col. 3, lines 1-9) the act of storing, “...storing a plurality of templates in a computer system comprises a memory having one or more locations, means for indexing one or more locations within the memory, template stored in at least one of the memory locations, initialization data associated with the template, key information (identifiers) associated with the template, and means coupling the memory to a central processing unit within the computer system.” It is inherent, to one skilled in the art, to associate each line of command line utility output (the modified template component), with a line identifier in system storage, using the means for indexing memory location. Thus, Buxton does disclose that the act of storing comprises “associating each line of command line utility output with a line identifier in the system storage.”

In conclusion: In the Buxton reference, a user creates / modifies custom template components (col. 11, lines 13-14) using a GUI or a command line interpreter (command line utility) (col. 8, lines 46-47). An editor module may selectively modify template component functionalities (col. 12, line 43). The template component is created with assistance of template storage dll (col. 13, lines 10-14), which ensures all additional registry keys and library types are created. Each template component comprises registry keys or subkeys and initialization data (col. 14, lines 26-29). The template component is given a name (identifier) (col. 12, line 3). The modified template component that is output by the GUI / command line interpreter (command line utility) may be stored (system storage) (col. 11, line 17 & col. 14, line 7). “When template

Art Unit: 2122

builder saves the template, it instructs the component, to save itself as an OLE structure storage file. Additional data to identify and register the component is also saved...(col. 14, lines 7-10)."

The means for indexing (col. 3, line 4) a template component are inherent, to one skilled in the art, by associating modified template component output with a line identifier in system storage.

Examiner clearly shows all facts and elements.

For the above reasons, it is believed that the rejections should be sustained, as reproduced below.

***Claim Rejections - 35 USC § 102***

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 1 - 22 are rejected under 35 U.S.C. 102(b) as being anticipated by U. S. Patent No. 6,182,279 to Buxton.

**Regarding claim 1, Buxton teaches:**

Note: Buxton, col. 8, lines 51-53, disclosed command line processes (command line utility output) as an alternative to a user interface.

-receiving an identifier. (Col. 12, lines 2-3: "User may enter a descriptive name (identifier) of the template..." Also, col. 9, lines 5-7, "Components register themselves...so that the OLE libraries are aware of the component's existence." As an example of "receiving an identifier", refer to col. 12, lines 18-21, "A menu...is presented upon selection of the Create Distribution Pack option and allows users to identify one or more templates to be packaged..." )



Art Unit: 2122

-receiving output from a command line utility. (Col. 13, line 8:) "Template builder creates templates..." Also see col. 2, lines 30-38 which recite, "A template builder utility allows a user to select a base component, modify the persistent data...and store (the received output) the modifications as a template in a predetermined format. The template builder utility further enables the user to package templates (received output from command line utility) in a template distribution package format which enables the templates to be distributed..."

-storing the command line utility output in a system storage at a location identified by the identifier. (Col. 11, lines 17-18, "...these templates may be stored to facilitate efficient distribution of templates to others...", "Col. 13, lines 10 – 14: "Template storage DLL (system storage) performs a number of ...storage/retrieval methods..." Also see col. 14, lines 7-12, "When template builder saves the template, it instructs the component, to save itself as an OLE structure storage file. Additional data to identify and register the component is also saved..."

Buxton disclosed a template builder utility (col. 2, lines 32-34) that allows user selected options to modify base components (OLE controls) and stores the modifications as templates.

**Regarding claim 2, Buxton teaches:**

-receiving an identifier that identifies one or more entries in a system registry database. (Fig. 2, item 205 and col. 13, lines 14-15, "...registry keys are created..." Also see col. 14, lines 29-59, "To facilitate loading of template onto another system...a number of registration key or subkey are included with template. Each template may have the keys 450A-I, as illustrated in Fig. 4C...Key 450H contains information indicating the name of the storage object in template storage file where initialization data...may be located...Key 450I contains information identifying the CLSID...)

Art Unit: 2122

**Regarding claim 3, Buxton teaches:**

-receiving a root key identifier. (Col. 11, line 2: "Most OLE object application information is stored in subkeys under the CSLID root key..." Also see col. 17, lines 35-41, "Component loader loads, verifies and checks the license of a component by replacing in registry the InProcessServer 32 entry, i.e. key 450A...and adding additional registry keys 450B-J, as previously described, that will let the component loader (receiving a root key identifier) then load the correct OLE control.")

**Regarding claim 4, Buxton teaches:**

-receiving a sub-key identifier. (Col. 11, line 2 and col. 14, line 31: To facilitate loading of template...a number of registration or subkey are included with template...")

**Regarding claim 5, Buxton teaches:**

-a WINDOWS operating system registry database. (Col. 4, line 49: "Operation of computer system is generally controlled...by operating system software, such as...Windows95...")

**Regarding claim 6, Buxton teaches:**

-receiving a system storage identifier. (Col. 12, lines 20-21, "...users identify...templates to be packaged..." Also for another example of receiving a system storage identifier, see col. 20, lines 42-45, "...relevant character string from the registry is converted to CLSID. The component loader (receives a system storage identifier) then calls the GetClassObject to retrieve the real component's class factory...")

**Regarding claim 7, Buxton teaches:**

Art Unit: 2122

-receiving an identifier indicating a system registry. (Col. 10, line 66 – col. 11, line 4: A CLSID identifies the functionality of an object class that can display...access to property values...A subkey is used by an OLE to find out information about the control.”)

**Regarding claim 8, Buxton teaches:**

-receiving an identifier indicating shared system memory. (Col. 8, lines 6-7: “OLE libraries (shared) comprise the set of system-level services in accordance with the OLE specification...”)

**Regarding claim 9, Buxton teaches:**

-shared system memory identifies a system clipboard memory. (Col. 11, line 6: “An FORMATETC...is an OLE data structure which acts in a generalized clipboard format...”)

**Regarding claims 10 and 11, Buxton teaches:**

-receiving output directly from the command line output utility.

-receiving output from the command line output utility through a subsequent command line output routine.

(Col. 2, lines 30-34, “A template builder utility allows a user to select a base component, modify the persistent data of the component, and store the modifications as a template in a predetermined format.” (storage receives modified templates) Also, col. 2, lines 34 – 38 and col. 13, lines 23-24: “The template builder utility further (subsequent command line output routine) enables the user to package (by receiving) templates (output for command line utility) in a template distribution package format which enables the templates to be distributed to other users...” Also see, col. 14, line 61-65, “...the Create Distribution Pack option...”)

**Regarding claim 12, Buxton teaches:**

Art Unit: 2122

-associating each line of command line utility output with a line identifier in the system storage.

(Col. 3, lines 1-9: Template storage with a means for indexing, including key information associated with the template. "...a memory having one or more locations, means for indexing one or more locations within the memory..." Also col. 13, lines 35-44, templates are stored with an enumerated decimal number: "Each template is stored in an ISTORE whose name is unique...and may have the form TEMPLEnnn, where nnn may be a decimal number.")

**Regarding claim 13, Buxton teaches:**

-setting each line identifier to a value corresponding to that line's position in the command line utility output. (Rejection of claim 12 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 13 is rejected under the same rational as claim 12.)

**Regarding claim 14, Buxton teaches:**

-setting a default value of the received identifier to equal the total number of command utility output lines stored in the system storage. (Rejection of claim 12 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 14 is rejected under the same rational as claim 12.)

**Regarding claim 15, Buxton teaches:**

Note: Buxton, col. 8, lines 51-53, disclosed command line processes as an alternative to a user interface.

-A program storage device (Col. 2, line 49- 52.)

Claim 15 contains limitations as recited in claim 1, therefore claim 15 is rejected under the same rational as claim 1.

**Regarding claim 16, Buxton teaches:**

Art Unit: 2122

-instructions to store command line utility output in an operating system registry database. (Col. 13, lines 10 – 15: “Template storage DLL (operating system registry database) performs a number of formatting and storage/retrieval methods...Template storage DLL ensures all additional registry keys...are created...”)

**Regarding claim 17, Buxton teaches:**

-instructions to store command line utility output in an operating system maintained volatile memory. (Fig. 1, item 110 (volatile) and col. 23, line 36 – col. 24, line 10: “A software implementation of the above described embodiment may comprise a series of computer instructions either fixed on a tangible medium...or transmittable to a computer system...such instructions may be stored using any memory technology ...on system ROM or fixed disk...”)

**Regarding claim 18, Buxton teaches:**

-instructions to receive one or more lines of output from the command line utility. (Fig. 2.)  
-instructions to store each of said one or more lines of output in the system storage. (Fig. 2, item 205, col. 14, lines 26-29: “The remainder of the operating system registry entries are generated by code (instructions to store) in the template storage DLL and are stored in both registry and the template.”)

**Regarding claim 19, Buxton teaches:**

-instructions to associate a unique identifier with each of the one or more lines of output stored in the system storage. (Col. 13, lines 40-41, “...name is unique to the machine for that template.”)

**Regarding claim 20, Buxton teaches:**

-instructions to set a value associated with the received identifier in the system storage equal to the number of lines of output stored in the system storage. (Rejection of claim 18 is incorporated

Art Unit: 2122

and further claim contains limitations as recited in claim 12. Therefore claim 20 is rejected under the same rational as claim 12.)

**Regarding claim 21, Buxton teaches:**

Note: Buxton, col. 8, lines 51-53, disclosed command line processes as an alternative to a user interface.

-a processor; a storage device coupled to the processor; the storage device having stored thereon a program having instructions to receive an identifier, receive output from a command line utility, and store the command line utility output in the system storage at a location identified by the identifier. (See fig. 1. Claim 21 contains limitations as recited in claim 1, therefore claim 21 is rejected under the same rational as claim 1.)

**Regarding claim 22, Buxton teaches:**

-program comprises a dynamic link library. (Fig. 2, item 205.)

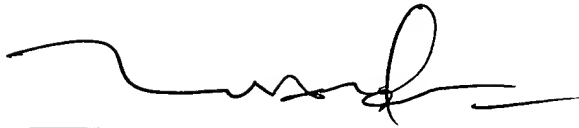
Respectfully submitted,

APPEAL PANEL

July 30, 2003

Conferees:

Mary Steelman, Examiner  
07/17/2003



Tuan Q. Dam, Supervisory Patent Examiner

COE F MILES  
TROP PRUNER HU & MILES P C  
8554 KATY FREEWAY  
SUITE 100  
HOUSTON, TX 77024



Antony Nguyen-Ba, Primary Examiner